# PCI DSS 4.0 Requirements and Testing Tools for AWS

Version 1.1

## 1. Requested Information from Client

| Information Required | Purpose | PCI DSS Requirement |
| --- | --- | --- |
| Network architecture diagrams | Validate network segmentation and security controls | 1.1.1, 1.2.1, 1.3.1 |
| AWS account access (read-only) | Review security configurations and controls | Multiple |
| List of in-scope systems/components | Determine assessment boundaries | 12.5.1, 12.5.2 |
| Inventory of CDE systems | Confirm all components are secured | 2.1.1, 11.2.1 |
| DNS architecture and records | Validate DNS security configurations | 1.3.1, 2.2.5, 4.1.1 |

## 2. Testing Procedures with Open-Source Tools

### Network Security Controls (Requirements 1.x)

| Testing Item | Primary Tool(s) | Open-Source Alternatives | Purpose |
| --- | --- | --- | --- |
| Security Group Rules Analysis | aws ec2 describe-security-groups | Prowler, ScoutSuite | Verify security group rules protect CDE (1.2.1-1.2.8) |
| Network Segmentation Verification | aws ec2 describe-subnets, aws ec2 describe-route-tables | Nmap, Zmap | Confirm isolation between CDE and untrusted networks (1.3.1-1.3.2) |
| Public-facing Security Controls | aws ec2 describe-instances, aws elb describe-load-balancers | Nmap, Nuclei | Validate protection of public-facing resources (1.5.1) |
| VPC Endpoint Controls | aws ec2 describe-vpc-endpoints | ScoutSuite | Verify service endpoint controls (1.3.1-1.3.2) |
| Network ACL Review | aws ec2 describe-network-acls | Prowler, CloudSploit | Verify network access control lists (1.2.1-1.2.8) |

#### Example Commands for Network Security Controls

| Testing Item | Example Command |
| --- | --- |
| Security Group Rules Analysis | aws ec2 describe-security-groups --filters Name=group-name,Values=\*cde\* --query "SecurityGroups[\*].{Name:GroupName,ID:GroupId,IngressRules:IpPermissions[\*]}" --output json |
| Network Segmentation Verification | aws ec2 describe-subnets --filters Name=tag:Environment,Values=PCI --query "Subnets[\*].{ID:SubnetId,CIDR:CidrBlock,VPC:VpcId,AZ:AvailabilityZone}" --output table |
| Public-facing Security Controls | aws ec2 describe-instances --filters Name=instance-state-name,Values=running --query "Reservations[\*].Instances[\*].{ID:InstanceId,PublicIP:PublicIpAddress,PrivateIP:PrivateIpAddress,Type:InstanceType,SecurityGroups:SecurityGroups[\*].GroupId}" --output table |
| VPC Endpoint Controls | aws ec2 describe-vpc-endpoints --query "VpcEndpoints[\*].{ID:VpcEndpointId,Service:ServiceName,VPC:VpcId,State:State,Policy:PolicyDocument}" --output json |
| Network ACL Review | aws ec2 describe-network-acls --query "NetworkAcls[\*].{ID:NetworkAclId,VPC:VpcId,Inbound:Entries[?Egress==\false`]}” –output json` |

### System Configuration Standards (Requirements 2.x)

| Testing Item | Primary Tool(s) | Open-Source Alternatives | Purpose |
| --- | --- | --- | --- |
| System Hardening Assessment | aws ec2 describe-instances, aws ssm describe-instance-information | Lynis, OpenSCAP | Verify systems are configured securely (2.2.1-2.2.7) |
| Administrative Access Review | aws iam list-roles, aws iam get-policy | CloudSploit, InSpec | Confirm encrypted admin access (2.3.1-2.3.2) |
| Default Configuration Analysis | aws rds describe-db-instances | AWS Config Rules, CloudSploit | Identify default settings and passwords (2.1.1) |
| EC2 Instance Configuration Analysis | aws ec2 describe-instances | InSpec, Prowler | Verify secure system component configurations (2.2.1-2.2.5) |
| AMI Configuration Review | aws ec2 describe-images | Amazon Inspector | Verify secure base images (2.2.1-2.2.5) |

#### Example Commands for System Configuration Standards

| Testing Item | Example Command |
| --- | --- |
| System Hardening Assessment | aws ssm describe-instance-information --query "InstanceInformationList[\*].{InstanceId:InstanceId,PlatformName:PlatformName,PlatformVersion:PlatformVersion,PingStatus:PingStatus}" --output table |
| Administrative Access Review | aws iam list-roles --query "Roles[?contains(RoleName, 'admin')].{RoleName:RoleName,CreateDate:CreateDate,Path:Path}" --output table |
| Default Configuration Analysis | aws rds describe-db-instances --query "DBInstances[\*].{DBIdentifier:DBInstanceIdentifier,Engine:Engine,MasterUsername:MasterUsername,MultiAZ:MultiAZ,Encrypted:StorageEncrypted}" --output table |
| EC2 Instance Configuration Analysis | aws ec2 describe-instances --query "Reservations[\*].Instances[\*].{ID:InstanceId,AMI:ImageId,Type:InstanceType,State:State.Name,LaunchTime:LaunchTime}" --output table |
| AMI Configuration Review | aws ec2 describe-images --owners self --query "Images[\*].{ImageId:ImageId,Name:Name,CreationDate:CreationDate,Public:Public}" --output table |

### Data Protection Mechanisms (Requirements 3.x, 4.x)

| Testing Item | Primary Tool(s) | Open-Source Alternatives | Purpose |
| --- | --- | --- | --- |
| Encryption Configuration Analysis | aws kms list-keys, aws s3 get-bucket-encryption | Prowler, CloudSploit | Verify encryption of stored CHD (3.4.1-3.5.1) |
| Key Management Review | aws kms describe-key | ScoutSuite, CloudMapper | Evaluate cryptographic key security (3.6.1-3.7.1) |
| Certificate Validation | aws acm list-certificates | SSLyze, testssl.sh | Confirm proper certificate implementation (4.1.1) |
| S3 Bucket Encryption | aws s3 get-bucket-encryption | S3Scanner, CloudSploit | Verify S3 bucket encryption settings (3.4.1) |
| EBS Volume Encryption | aws ec2 describe-volumes | Prowler | Verify EBS volume encryption (3.4.1) |

#### Example Commands for Data Protection Mechanisms

| Testing Item | Example Command |
| --- | --- |
| Encryption Configuration Analysis | aws kms list-keys --query "Keys[\*].KeyId" --output text \| xargs -I {} aws kms describe-key --key-id {} --query "KeyMetadata.{KeyId:KeyId,Description:Description,Enabled:Enabled,KeyUsage:KeyUsage}" --output table |
| Key Management Review | aws kms list-aliases --query "Aliases[?contains(AliasName, 'pci') \|\| contains(AliasName, 'payment')].TargetKeyId" --output text \| xargs -I {} aws kms get-key-rotation-status --key-id {} |
| Certificate Validation | aws acm list-certificates --query "CertificateSummaryList[\*].{CertificateArn:CertificateArn,DomainName:DomainName,Status:Status}" --output table |
| S3 Bucket Encryption | aws s3api list-buckets --query "Buckets[\*].Name" --output text \| xargs -I {} aws s3api get-bucket-encryption --bucket {} 2>/dev/null \|\| echo "Bucket {} not encrypted" |
| EBS Volume Encryption | aws ec2 describe-volumes --query "Volumes[\*].{VolumeId:VolumeId,State:State,Encrypted:Encrypted,Size:Size,AZ:AvailabilityZone}" --output table |

### Access Control Assessment (Requirements 7.x, 8.x)

| Testing Item | Primary Tool(s) | Open-Source Alternatives | Purpose |
| --- | --- | --- | --- |
| IAM Policy Analysis | aws iam get-account-authorization-details | IAM Access Analyzer, Prowler | Verify principle of least privilege (7.1.1-7.3.3) |
| Access Path Mapping | aws iam list-roles, aws iam list-users | CloudMapper, PMapper | Identify access paths to CHD systems (7.2.4) |
| Authentication Mechanisms | aws iam list-virtual-mfa-devices | Prowler, ScoutSuite | Review MFA and authentication controls (8.3.1-8.6.3) |
| Password Policy Review | aws iam get-account-password-policy | CloudSploit, Prowler | Verify password policy compliance (8.6.1-8.6.3) |
| Cross-Account Access Review | aws organizations list-accounts | CloudMapper, Prowler | Review cross-account access (7.3.1-7.3.3) |

#### Example Commands for Access Control Assessment

| Testing Item | Example Command |
| --- | --- |
| IAM Policy Analysis | aws iam get-account-authorization-details --filter IAM-USER --query "UserDetailList[\*].{UserName:UserName,AttachedManagedPolicies:AttachedManagedPolicies[\*].PolicyName}" --output json |
| Access Path Mapping | aws iam list-users --query "Users[\*].{UserName:UserName,UserId:UserId,CreateDate:CreateDate}" --output table && aws iam list-roles --query "Roles[\*].{RoleName:RoleName,CreateDate:CreateDate}" --output table |
| Authentication Mechanisms | aws iam list-virtual-mfa-devices --query "VirtualMFADevices[\*].{User:User.UserName,SerialNumber:SerialNumber,EnableDate:EnableDate}" --output table |
| Password Policy Review | aws iam get-account-password-policy --query "PasswordPolicy.{MinimumPasswordLength:MinimumPasswordLength,RequireSymbols:RequireSymbols,RequireNumbers:RequireNumbers,RequireUppercaseCharacters:RequireUppercaseCharacters,RequireLowercaseCharacters:RequireLowercaseCharacters,PasswordReusePrevention:PasswordReusePrevention,MaxPasswordAge:MaxPasswordAge}" --output table |
| Cross-Account Access Review | aws organizations list-accounts --query "Accounts[\*].{Id:Id,Name:Name,Status:Status,Email:Email}" --output table |

### Vulnerability Management (Requirements 6.x, 11.x)

| Testing Item | Primary Tool(s) | Open-Source Alternatives | Purpose |
| --- | --- | --- | --- |
| WAF Configuration Review | aws waf list-web-acls | OWASP ZAP, Nuclei | Verify protection of web applications (6.4.1-6.4.2) |
| System Component Analysis | aws ecr describe-images | Trivy, Clair | Identify outdated software components (6.3.1-6.3.3) |
| Patch Management Review | aws ssm describe-patch-baselines | OpenVAS, Prowler | Verify patch status and processes (6.3.3) |
| Vulnerability Scanning | aws inspector list-findings | OWASP ZAP, Nikto | Identify vulnerabilities in systems (11.3.1-11.3.2) |
| Guard Duty Findings | aws guardduty list-findings | Prowler, CloudSploit | Review security findings (11.4.1-11.4.5) |

#### Example Commands for Vulnerability Management

| Testing Item | Example Command |
| --- | --- |
| WAF Configuration Review | aws wafv2 list-web-acls --scope REGIONAL --query "WebACLs[\*].{Name:Name,Id:Id,ARN:ARN}" --output table |
| System Component Analysis | aws ecr describe-repositories --query "repositories[\*].{Name:repositoryName,URI:repositoryUri,Created:createdAt}" --output table |
| Patch Management Review | aws ssm describe-patch-baselines --query "BaselineIdentities[\*].{Name:BaselineName,ID:BaselineId,OperatingSystem:OperatingSystem,Description:BaselineDescription}" --output table |
| Vulnerability Scanning | aws inspector2 list-findings --filter "findingStatus={comparison=EQUALS,value=ACTIVE}" --max-results 10 --query "findings[\*].{Title:title,Severity:severity,ResourceId:resources[0].id}" --output table |
| Guard Duty Findings | aws guardduty list-detectors --query "DetectorIds" --output text \| xargs -I {} aws guardduty list-findings --detector-id {} --finding-criteria '{"Criterion":{"severity":{"Eq":["8","9"]}}}' |

### Logging and Monitoring (Requirements 10.x, 11.x)

| Testing Item | Primary Tool(s) | Open-Source Alternatives | Purpose |
| --- | --- | --- | --- |
| CloudTrail Configuration | aws cloudtrail describe-trails | CloudSploit, Prowler | Verify audit logging is enabled (10.2.1-10.2.2) |
| Log Configuration Review | aws logs describe-log-groups | ELK Stack, Graylog | Confirm proper log settings and retention (10.5.1-10.7.3) |
| CloudWatch Alarms Review | aws cloudwatch describe-alarms | Grafana, Prometheus | Verify continuous monitoring (11.4.1-11.4.5) |
| Log Content Validation | aws logs filter-log-events | Logstash, Loki | Confirm logging of required events (10.2.1-10.2.2) |
| VPC Flow Logs Review | aws ec2 describe-flow-logs | Zeek, Suricata | Verify network traffic logging (10.2.2, 10.3.2) |

#### Example Commands for Logging and Monitoring

| Testing Item | Example Command |
| --- | --- |
| CloudTrail Configuration | aws cloudtrail describe-trails --query "trailList[\*].{Name:Name,IsMultiRegion:IsMultiRegionTrail,LoggingEnabled:IsLogging,HomeRegion:HomeRegion}" --output table |
| Log Configuration Review | aws logs describe-log-groups --query "logGroups[\*].{LogGroupName:logGroupName,RetentionInDays:retentionInDays,StoredBytes:storedBytes}" --output table |
| CloudWatch Alarms Review | aws cloudwatch describe-alarms --state-value ALARM --query "MetricAlarms[\*].{Name:AlarmName,Metric:MetricName,Namespace:Namespace,State:StateValue}" --output table |
| Log Content Validation | aws logs filter-log-events --log-group-name /aws/cloudtrail --filter-pattern "{$.eventName = ConsoleLogin}" --limit 5 |
| VPC Flow Logs Review | aws ec2 describe-flow-logs --query "FlowLogs[\*].{Id:FlowLogId,LogDestination:LogDestination,ResourceId:ResourceId,TrafficType:TrafficType,Status:FlowLogStatus}" --output table |

### DNS Security Assessment

| Testing Item | Primary Tool(s) | Open-Source Alternatives | Purpose |
| --- | --- | --- | --- |
| Route 53 Configuration Analysis | aws route53 list-hosted-zones | DNSRecon, Fierce | Verify proper DNS settings for CDE components (1.2.1, 2.2.5) |
| DNS Security Extensions | aws route53 get-hosted-zone | DNSViz, delv | Confirm DNSSEC implementation if applicable (4.1.1) |
| Zone Transfer Testing | aws route53 list-resource-record-sets | DNSenum, dig | Identify zone transfer vulnerabilities (1.3.1, 11.3.1) |
| DNS Record Enumeration | aws route53 list-resource-record-sets | Amass, Sublist3r | Map DNS infrastructure and verify segmentation (1.3.1) |
| Domain Security Analysis | aws route53domains list-domains | dnstwist, DNSTwist | Detect typosquatting & domain security issues (6.4.1, 11.3.1) |

#### Example Commands for DNS Security Assessment

| Testing Item | Example Command |
| --- | --- |
| Route 53 Configuration Analysis | aws route53 list-hosted-zones --query "HostedZones[\*].{Name:Name,Id:Id,Private:Config.PrivateZone}" --output table |
| DNS Security Extensions | aws route53 list-hosted-zones --query "HostedZones[\*].Id" --output text \| xargs -I {} aws route53 get-dnssec --hosted-zone-id {} 2>/dev/null \|\| echo "DNSSEC not enabled" |
| Zone Transfer Testing | aws route53 list-hosted-zones --query "HostedZones[\*].Id" --output text \| head -1 \| xargs -I {} aws route53 list-resource-record-sets --hosted-zone-id {} --query "ResourceRecordSets[?Type=='NS']" |
| DNS Record Enumeration | aws route53 list-hosted-zones --query "HostedZones[\*].Id" --output text \| head -1 \| xargs -I {} aws route53 list-resource-record-sets --hosted-zone-id {} --query "ResourceRecordSets[\*].{Name:Name,Type:Type,TTL:TTL}" --output table |
| Domain Security Analysis | aws route53domains list-domains --query "Domains[\*].{DomainName:DomainName,Expiry:Expiry,AutoRenew:AutoRenew}" --output table |

## 3. Required AWS User Account for Assessment

For PCI DSS 4.0 assessment, a regular AWS IAM user account with appropriate read-only permissions is required. This account can be used to log in through the AWS Management Console and AWS CLI normally.

### Required User Account Permissions

The following AWS managed policies should be assigned to the user account for comprehensive assessment capabilities:

| Policy | Purpose | PCI DSS Requirements |
| --- | --- | --- |
| ReadOnlyAccess | Base read-only access to most AWS resources | Multiple |
| SecurityAudit | Access to security-specific information | 7.1, 7.2, 8.1, 11.3 |
| AmazonInspector2ReadOnlyAccess | Access to Inspector vulnerability findings | 6.3, 11.3 |
| AmazonGuardDutyReadOnlyAccess | Access to GuardDuty security findings | 10.6, 11.4 |
| CloudWatchReadOnlyAccess | Access to monitoring and metrics | 10.4, 10.6, 11.5 |
| AWSCloudTrailReadOnlyAccess | Access to audit log data | 10.2, 10.3, 10.5 |
| AWSConfigUserAccess | Access to AWS Config compliance data | 1.2.7, 2.2.1, 6.3.3 |
| AmazonVPCReadOnlyAccess | Access to VPC configurations | 1.2, 1.3, 1.4 |
| IAMReadOnlyAccess | Access to IAM configurations | 7.1, 7.2, 8.1, 8.3 |
| AmazonS3ReadOnlyAccess | Access to S3 bucket configurations | 3.4, 9.4, 10.5 |
| AWSSecurityHubReadOnlyAccess | Access to Security Hub findings | 6.3, 11.3, 11.4 |

### Custom PCI Assessor Policy

Instead of assigning multiple predefined policies, we recommend creating a single custom policy with exactly the permissions needed for the assessment. This approach follows the principle of least privilege (Requirement: 7.2.1) while ensuring access to all necessary resources for a complete PCI DSS assessment:

{  
 "Version": "2012-10-17",  
 "Statement": [  
 {  
 "Effect": "Allow",  
 "Action": [  
 "acm:List\*",  
 "acm:Describe\*",  
 "cloudtrail:Describe\*",  
 "cloudtrail:Get\*",  
 "cloudtrail:List\*",  
 "cloudtrail:LookupEvents",  
 "cloudwatch:Describe\*",  
 "cloudwatch:Get\*",  
 "cloudwatch:List\*",  
 "config:BatchGetResourceConfig",  
 "config:Get\*",  
 "config:List\*",  
 "config:SelectResourceConfig",  
 "ec2:Describe\*",  
 "ecr:Describe\*",  
 "ecr:List\*",  
 "ecs:Describe\*",  
 "ecs:List\*",  
 "eks:Describe\*",  
 "eks:List\*",  
 "elasticloadbalancing:Describe\*",  
 "events:Describe\*",  
 "events:List\*",  
 "guardduty:Get\*",  
 "guardduty:List\*",  
 "iam:GenerateCredentialReport",  
 "iam:GenerateServiceLastAccessedDetails",  
 "iam:Get\*",  
 "iam:List\*",  
 "iam:SimulatePrincipalPolicy",  
 "inspector2:List\*",  
 "inspector2:Describe\*",  
 "inspector2:Get\*",  
 "kms:Describe\*",  
 "kms:Get\*",  
 "kms:List\*",  
 "lambda:GetFunction",  
 "lambda:GetFunctionConfiguration",  
 "lambda:List\*",  
 "logs:Describe\*",  
 "logs:FilterLogEvents",  
 "logs:Get\*",  
 "logs:StartQuery",  
 "logs:StopQuery",  
 "logs:TestMetricFilter",  
 "macie2:Describe\*",  
 "macie2:Get\*",  
 "macie2:List\*",  
 "organizations:Describe\*",  
 "organizations:List\*",  
 "rds:Describe\*",  
 "route53:Get\*",  
 "route53:List\*",  
 "route53domains:GetDomainDetail",  
 "route53domains:List\*",  
 "s3:Get\*",  
 "s3:List\*",  
 "secretsmanager:List\*",  
 "secretsmanager:Describe\*",  
 "securityhub:BatchGetSecurityControls",  
 "securityhub:Get\*",  
 "securityhub:List\*",  
 "ssm:Describe\*",  
 "ssm:Get\*",  
 "ssm:List\*",  
 "shield:Describe\*",  
 "shield:Get\*",  
 "shield:List\*",  
 "sns:Get\*",  
 "sns:List\*",  
 "waf:Get\*",  
 "waf:List\*",  
 "wafv2:Get\*",  
 "wafv2:List\*"  
 ],  
 "Resource": "\*"  
 }  
 ]  
}

### Creating and Assigning the Custom Policy

# Create a local JSON file with the policy  
cat > pci-assessor-policy.json << 'EOF'  
{  
 "Version": "2012-10-17",  
 "Statement": [  
 {  
 "Effect": "Allow",  
 "Action": [  
 "acm:List\*",  
 "acm:Describe\*",  
 "cloudtrail:Describe\*",  
 "cloudtrail:Get\*",  
 "cloudtrail:List\*",  
 "cloudwatch:Describe\*",  
 "cloudwatch:Get\*",  
 "cloudwatch:List\*",  
 "config:Get\*",  
 "config:List\*",  
 "ec2:Describe\*",  
 "ecr:Describe\*",  
 "ecr:List\*",  
 "ecs:Describe\*",  
 "ecs:List\*",  
 "eks:Describe\*",  
 "eks:List\*",  
 "elasticloadbalancing:Describe\*",  
 "guardduty:Get\*",  
 "guardduty:List\*",  
 "iam:Get\*",  
 "iam:List\*",  
 "inspector2:List\*",  
 "inspector2:Describe\*",  
 "kms:Describe\*",  
 "kms:Get\*",  
 "kms:List\*",  
 "lambda:List\*",  
 "logs:Describe\*",  
 "logs:FilterLogEvents",  
 "logs:Get\*",  
 "organizations:List\*",  
 "rds:Describe\*",  
 "route53:Get\*",  
 "route53:List\*",  
 "route53domains:List\*",  
 "s3:Get\*",  
 "s3:List\*",  
 "secretsmanager:List\*",  
 "secretsmanager:Describe\*",  
 "securityhub:Get\*",  
 "securityhub:List\*",  
 "ssm:Describe\*",  
 "ssm:Get\*",  
 "ssm:List\*",  
 "waf:Get\*",  
 "waf:List\*",  
 "wafv2:Get\*",  
 "wafv2:List\*"  
 ],  
 "Resource": "\*"  
 }  
 ]  
}  
EOF  
  
# Create the custom policy  
aws iam create-policy \  
 --policy-name PCI\_DSS\_Assessor \  
 --policy-document file://pci-assessor-policy.json  
  
# Create a user for the assessment  
aws iam create-user --user-name pci\_assessor  
  
# Attach the policy to the user  
aws iam attach-user-policy \  
 --user-name pci\_assessor \  
 --policy-arn arn:aws:iam::ACCOUNT\_ID:policy/PCI\_DSS\_Assessor

### Account Setup Verification

After policy assignment, verify the user has proper access by testing access to critical services needed for PCI DSS assessment:

# Create access key for the assessor account  
aws iam create-access-key --user-name pci\_assessor  
  
# Configure AWS CLI with the new credentials  
aws configure  
  
# Verify access to network security controls (Requirement 1.x)  
aws ec2 describe-security-groups --region us-east-1 --max-items 1  
aws ec2 describe-network-acls --region us-east-1 --max-items 1  
aws ec2 describe-subnets --region us-east-1 --max-items 1  
  
# Verify access to system configuration data (Requirement 2.x)  
aws ec2 describe-instances --region us-east-1 --max-items 1  
aws rds describe-db-instances --region us-east-1 --max-items 1  
  
# Verify access to encryption information (Requirement 3.x)  
aws kms list-keys --region us-east-1 --max-items 1  
aws s3 get-bucket-encryption --bucket <sample-bucket-name> 2>/dev/null || echo "Access verified"  
  
# Verify access to access control information (Requirements 7.x, 8.x)  
aws iam list-users --max-items 1  
aws iam get-account-password-policy 2>/dev/null || echo "Access verified"  
  
# Verify access to logging and monitoring data (Requirement 10.x)  
aws cloudtrail describe-trails --region us-east-1  
aws logs describe-log-groups --region us-east-1 --max-items 1  
aws cloudwatch describe-alarms --region us-east-1 --max-items 1  
  
# Verify access to vulnerability management data (Requirements 6.x, 11.x)  
aws inspector2 list-findings --region us-east-1 --max-items 1 2>/dev/null || echo "Access verified"  
aws securityhub list-findings --region us-east-1 --max-items 1 2>/dev/null || echo "Access verified"  
aws guardduty list-findings --region us-east-1 --max-items 1 2>/dev/null || echo "Access verified"  
  
# List policies attached to the user to confirm proper setup  
aws iam list-attached-user-policies --user-name pci\_assessor

This comprehensive testing approach ensures the assessment account has read access to all systems required for PCI DSS 4.0 compliance validation, covering each major requirement category.

### Notes

1. Testing will be conducted from a virtual machine running Ubuntu 24.04 with the following specifications:
   * 8GB RAM
   * 2 CPUs
   * 50GB SSD
2. No tools will be installed on any AWS systems
3. All testing is non-intrusive and performed with read-only AWS account access
4. Primary assessment will be performed using AWS CLI tools
5. All tools mentioned are open source and free to use
6. This assessment process is for PCI DSS 4.0 compliance validation, not penetration testing
7. The AWS CLI commands provide the core assessment capabilities, with open source tools for additional validation
8. The IAM permissions provided align with PCI DSS 4.0 Requirement 7.2.1 for least privilege access
9. AWS Config Rules and AWS Security Hub can provide additional automated checks for PCI DSS compliance
10. For Requirement 11.3.2, an ASV-approved scanning vendor is still required for external vulnerability scans
11. AWS native services like Amazon Inspector, GuardDuty, and Security Hub can assist with meeting Requirements 6.3.1, 10.7.2, and 11.4.1, but do not replace all required controls